1

48p.

JPL Ext Pub. 794

Code-5

NASA CR-50981

MEMORANDUM NO. 30-1

MULTIDIMENSIONAL LEAST-SQUARES CURVE FITTING AND SEPARATION OF VARIABLES

ON 11873

H. F. LESH Feb. 27, 1959

7 rfx

index

(NASA CR-50981; JPL-MEMO-30-1)

[2] say

JET PROPULSION LABORATORY
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA
FEBRUARY 27, 1959

1

National Aeronautics and Space Administration
Contract No. NASw-6

Memorandum No. 30-1

MULTIDIMENSIONAL LEAST-SQUARES CURVE FITTING AND SEPARATION OF VARIABLES

H. F. Lesh

William Hoover

William Hoover, *Chief*
Applied Mathematics Section

JET PROPULSION LABORATORY
California Institute of Technology
Pasadena, California
February 27, 1959

PREFACE

Portions of the following report were originated under studies conducted for the Department of Army Ordnance Corps under Contract No. DA-04-495-Ord 18. Such studies are now conducted for the National Aeronautics and Space Administration under Contract No. NASw-6.

CONTENTS

TABLES

FIGURES

ABSTRACT

The results of investigations into certain areas of curve fitting are presented. The derivation of the normal equations in two dimensions is discussed and an easily generalizable way of viewing the structure of the normal equations is given. A new method is presented for the approximate separation of variables in an empirically determined function. The need for double precision calculation is explained, and the routines for curve fitting which are available at the Jet Propulsion Laboratory are summarized.

I. INTRODUCTION

When least-squares curve fits of orders greater than 4 began to be run on the general purpose digital computer, it was noticed that the fits, far from getting better as the order of the polynomial increased, were actually getting worse. This led to an investigation of the analysis of curve fitting and to the writing of a double-precision curve-fitting routine to handle curve fits up to orders as great as 10 or 11. Later, the need arose for routines to fit polynomials in two or three dimensions, and for a routine to separate variables in two or three dimensions.

The purpose of this Report is to make known the results of investigations into the problems of multidimensional curve fitting and separation of variables, and to outline the nature of the programs available for doing these kinds of work.

In particular, this Report:

1. Summarizes portions of the curve-fitting theory.
2. Presents an easily generalizable method of setting up normal equations in any number of dimensions.
3. Presents some new results on separation of variables.
4. Explains the need for double-precision calculation.
5. Summarizes the available routines for curve fitting.

II. THE GENERAL CURVE-FITTING PROBLEM

There are two common motivations for doing curve fitting. In some cases a set of data may be known to contain observational errors of a random type, and it is desired to find the most likely values of the observed function. By making the assumptions that the underlying trend is an mth-order polynomial and that the noise is uncorrelated and normally distributed with mean zero, one is led directly to the standard normal equations of the least-squares process (see Ref. 1). A proof of this statement is contained in Appendix A.

In other cases a function may be known exactly on a discreet point set (or with sufficient precision so that one can consider it known exactly), and it may be desired to find a calculational algorithm to approximate the original function. In this case it may be desirable to use some measure of error other than that of the sum of the squares of the point wise errors, such as the magnitude of the largest absolute deviation.[1] These other measures of error are generally rather hard to handle, however, and since the least-squares measure of error leads to very simple analytic results, it is usually used.

A. Derivation of the Normal Equations

The ordinary one-dimensional least-squares analysis with powers of x is well-known (Ref. 3), and there is no need to repeat it here. The general analysis in n dimensions, although it presents no difficulty, is seldom seen in the literature. The notation of two dimensions will be used to present this theory since it contains enough structure to make clear the extension to n dimensions and because the notation necessary in n dimensions is so complex as to obscure the argument.

We wish to determine the a_i in the polynomial

$$P(x,y) = a_0 x^{\lambda_0} y^{\mu_0} + \cdots + a_m x^{\lambda_m} y^{\mu_m} .$$

[1] For examples of approximations made using the magnitude of the largest error as a criterion of "goodness of fit" see Ref. 2.

in such a way that E is a minimum where:

$$E = \sum_{i=1}^{n} [F(x_i, y_i) - P(x_i, y_i)]^2$$

The conditions which have to be satisfied for a minimum are:

$$\frac{\partial E}{\partial a_\nu} = 0 \qquad \nu = 0, \cdots, m$$

But

$$\frac{\partial E}{\partial a_\nu} = 2 \sum_{i=1}^{n} (F - P)\, x_i^{\lambda_\nu} y_i^{\mu_\nu}$$

So that the normal equations become

$$a_0 \sum_i x_i^{\lambda_\nu+\lambda_0} y_i^{\mu_\nu+\mu_0} + \cdots + a_m \sum_i x_i^{\lambda_\nu+\lambda_m} y_i^{\mu_\nu+\mu_m} = \sum_i F(x_i, y_i)\, x_i^{\lambda_\nu} y_i^{\mu_\nu}$$

$$\nu = 0, \cdots, m$$

or

$$Ax = b$$

where

$$A = \begin{pmatrix} \sum\limits_i x_i^{\lambda_0+\lambda_0} \, y_i^{\mu_0+\mu_0} & \cdots & \sum x_i^{\lambda_0+\lambda_m} \, y_i^{\mu_0+\mu_m} \\ \cdot & & \\ \cdot & & \\ \cdot & & \\ \sum\limits_i x_i^{\lambda_m+\lambda_0} \, y_i^{\mu_m+\mu_0} & \cdots & \sum x_i^{\lambda_m+\lambda_m} \, y_i^{\mu_m+\mu_m} \end{pmatrix}$$

$$x = \begin{pmatrix} a_0 \\ \cdot \\ \cdot \\ \cdot \\ a_m \end{pmatrix}$$

and

$$b = \begin{matrix} \sum\limits_i F(x_i, y_i) \; x_i^{\lambda_0} \, y_i^{\mu_0} \\ \cdot \\ \cdot \\ \cdot \\ \sum\limits_i F(x_i, y_i) \; x_i^{\lambda_m} \, y_i^{\mu_m} \end{matrix}$$

B. A General Method for Writing the Normal Equations

An easily generalizable way of looking at the structure of the matrix A may be seen by defining an exponent vector to be the set of integers (λ, μ) which appear as exponents of x and y in a single term of the approximating polynomial. Now let the polynomial be

$$P(x,y) = a_0 x^{\lambda_0} y^{\mu_0} + \cdots a_m x^{\lambda_m} y^{\mu_m}$$

so that the exponent vectors are

$$\begin{array}{c} (\lambda_0, \mu_0) \\ (\lambda_1, \mu_1) \\ \cdot \\ \cdot \\ \cdot \\ (\lambda_m, \mu_m) \end{array}$$

Now form the addition table of the exponent vectors as follows:

$$\begin{array}{ccccc}
 & (\lambda_0, \mu_0) & (\lambda_1, \mu_1) & \cdots & (\lambda_m, \mu_m) \\
(\lambda_0, \mu_0) & (u, v)_{00} & (u, v)_{01} & \cdots & (u, v)_{0m} \\
(\lambda_1, \mu_1) & (u, v)_{10} & (u, v)_{11} & \cdots & (u, v)_{1m} \\
\vdots & \vdots & \vdots & & \vdots \\
(\lambda_m, \mu_m) & (u, v)_{m0} & (u, v)_{m1} & \cdots & (u, v)_{mm}
\end{array}$$

where $(u, v)_{kl} = (\lambda_k + \lambda_l, \mu_k + \mu_l)$. Now let

$$\sum_i x_i^u y_i^v \rightarrow (u, v)$$

symbolize the operation of replacing the pair (u, v) by the number

$$\sum_i x_i^u \; y_i^v$$

Then, making the substitution

$$\sum_i x_i^{\lambda_k + \lambda_l} \; y_i^{\mu_k + \mu_l} \rightarrow (u, v)_{kl} \qquad k = 1, \cdots, m$$

the elements of the addition table become exactly the elements of the matrix A for the chosen polynomial. For example, let $P(x,y) = a_0 + a_1 x^2 + a_2 x^3 y^5$. Then the exponent vectors are

(0,0)
(2,0)
(3,5)

and the addition table is as follows:

	0,0	2,0	3,5
0,0	0,0	2,0	3,5
2,0	2,0	4,0	5,5
3,5	3,5	5,5	6,10

Making the replacement

$$\sum_i x_i^u y_i^v \rightarrow (u, v)$$

in the body of the table gives

$$\begin{array}{ccc}
n & \sum_i x_i^2 & \sum_i x_i^3 y_i^5 \\
\sum_i x_i^2 & \sum_i x_i^4 & \sum_i x_i^5 y_i^5 \\
\sum_i x_i^3 y_i^5 & \sum_i x_i^6 y_i^5 & \sum_i x_i^6 y_i^{10}
\end{array}$$

which is exactly the array of the matrix A for the given Polynomial.

Now it is easy to see how generalization to three dimensions occurs. The addition table looks like the following:

$$\begin{array}{cccc}
 & (\lambda_0, \mu_0, \nu_0) & \cdots & (\lambda_m, \mu_m, \nu_m) \\
(\lambda_0, \mu_0, \nu_0) & (u, v, w)_{00} & \cdots & (u, v, w)_{0m} \\
\vdots & \vdots & & \vdots \\
(\lambda_m, \mu_m, \nu_m) & (u, v, w)_{m0} & \cdots & (u, v, w)_{mm}
\end{array}$$

where $(u, v, w)_{kl} = (\lambda_k + \lambda_l, \mu_k + \mu_l, \nu_k + \nu_l)$ and the necessary replacement is:

$$\sum_i x_i^u y_i^v z_i^w \rightarrow (u, v, w)$$

Note that in the one-dimensional case the exponent vectors are just single integers and the replacement becomes:

$$\sum_i x_i^u \rightarrow u$$

For the special one-dimensional case where $P(x) = a_0 + \cdots + a_m x^m$ the addition table is:

	0	1	2	$\cdots$	m
0	0	1	2	$\cdots$	m
1	1	2	3	$\cdots$	$m+1$
2	2	3	4	$\cdots$	$m+2$
.	.				.
.	.				.
.	.				.
m	m	$m+1$	$m+2$	$\cdots$	$2m$

which becomes after replacement:

$$\begin{array}{ccccc}
n & \sum\limits_i x_i & \sum\limits_i x_i^2 & \cdots & \sum\limits_i x_i^m \\
\sum\limits_i x_i & \sum\limits_i x_i^2 & \sum\limits_i x_i^3 & \cdots & \sum\limits_i x_i^{m+1} \\
\cdot & & & & \cdot \\
\cdot & & & & \cdot \\
\cdot & & & & \cdot \\
\sum\limits_i x_i^m & \sum\limits_i x_i^{m+1} & \sum\limits_i x_i^{m+2} & \cdots & \sum\limits_i x_i^{2m}
\end{array}$$

which is the familiar matrix of the normal equations.

C. Orthogonal Polynomials

Suppose we have a set of polynomials $P_i(x)$ ($i = 0, \cdots, m$) with the property that $p_i(x)$ is a polynomial in x of degree i with a non-zero leading coefficient. Since any polynomial of degree m is uniquely representable as a linear combination $c_0p_0(x) + \cdots + c_mp_m(x)$, we can consider the problem of approximating an empirically determined function $F(x_i)$ $i = 1, \cdots, n$ by such a linear combination of the $P_i(x)$. In this case the sum of squares of the errors is

$$E = \sum_{i=1}^{n} \{F(x_i) - [c_0 + \cdots + c_mP_m(x)]\}^2$$

and taking partial derivatives with respect to the c_ν yields the normal equations

$$\sum_{i=1}^{n} [F(x_i) - (c_0 + \cdots + c_mP_m(x)] \, P_\nu(x) = 0 \qquad \nu = 0, \cdots, m$$

The normal matrix is:

$$\begin{matrix} (P_0,P_0) & (P_0,P_1) & \cdots & (P_0,P_m) \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ (P_m,P_0) & (P_m,P_1) & \cdots & (P_m,P_m) \end{matrix}$$

where (P_μ,P_ν) is the inner product

$$\sum_{i=1}^{n} P_\mu(x_i)\, P_\nu(x_i)$$

If a set of polynomials can be obtained with the property that $(P_\mu,P_\nu) = 0$ if $\mu \neq \nu$, then only the diagonal terms will remain in the normal matrix and the solution to the system will be:

$$a_\nu = \frac{(F,P_\nu)}{(P_\nu,P_\nu)} \qquad \nu = 0, \cdots, m$$

Such a set of polynomials is called an orthogonal set with respect to the inner product

$$\sum_{i=1}^{n} P_\mu(x_i)\, P_\nu(x_i)$$

and can be easily generated using the following recursion scheme:

$$P_0(x) = 1$$

$$p_1(x) = xp_0(x) - \alpha_0 p_0(x)$$

$$p_2(x) = xp_1(x) - \alpha_1 p_1(x) - \beta_0 p_0(x)$$

$$\vdots$$

$$P_{n+1}(x) = xP_n(x) - \alpha_n P_n(x) - \beta_{n-1} P_{n-1}(x)$$

where

$$\alpha_n = \frac{(xP_n,P_n)}{(P_n,P_n)}$$

$$\beta_n = \frac{(xP_{n+1},P_n)}{(P_n,P_n)}$$

A proof that these polynomials have the property that $(P_\mu,P_\nu) = 0$ if $\mu \neq \nu$ is contained in Appendix B.

The advantages of orthogonal polynomials are as follows:

1. There is no accuracy problem in obtaining either the orthogonal polynomials or the coefficients of the orthogonal polynomials for a particular fit.

2. Once having obtained the polynomials, they may be used to fit many different functions as long as the x_i remain fixed.

3. The F-test is a test for determining what order polynomial should be used to fit a given set of data. The test consists of calculating

$$\sigma_k = \frac{1}{n-k-1} \sum_{i=1}^{n} [f(x_i) - P_k(x_i)]^2$$

for $k = 1, 2, \cdots$ and testing the null hypothesis that σ_n^2 and σ_{n+1}^2 are sample variances from populations with the same variance (Ref. 4). When the null hypothesis is accepted, the smaller order fit is accepted. Using orthogonal polynomials, the reduction in σ^2 can be calculated without doing the fit all over again (Ref.˙4). If the P_k are orthogonal polynomials, the values of σ_k^2 for increasing k can be calculated using a recursion formula, while, if the P_k are powers of x, the whole operation must be repeated, or a complicated computer program must be developed.

The disadvantages of orthogonal polynomials are as follows:

1. If an approximation algorithm is being sought, the needed quantities are the coefficients of the powers of x, since the powers of x are much easier to compute than the values of the orthogonal polynomials. But in trying to recover the coefficients of the powers of x, the same accuracy problem is encountered which arises in trying to solve the normal equations.

2. In using orthogonal polynomials one is forced to use all powers of x less than or equal to the degree of the fit. This may be undesireable, e.g., only the odd powers may be wanted.

3. There are statements in the literature that orthogonal polynomials can be extended to more than one dimension, but little seems to have been done along these lines.

For further details on the use of orthogonal polynomials see Ref. 5.

III. A NEW TECHNIQUE FOR THE APPROXIMATE SEPARATION OF VARIABLES

The following analysis was done because of a suspicion that certain functions of two variables which arise in the approximation of missile-guidance parameters were very nearly separable. Separable is used here to mean that the function $f(x,y)$ can be written as the product of a function of x only by a function of y only.

It was felt that perhaps, for the cases in which a function was nearly separable, a much better fit to the data might be obtained by fitting $f_1(x)$ and $f_2(x)$ separately than by fitting $f(x,y)$ directly.

Given an empirically determined function $f(x,y)$ which is suspected of being nearly separable in the sense that $f(x,y) = f_1(x)\, f_2(y) + E(x,y)$ where the $E(x,y)$ are all very small compared to the $f(x,y)$. Suppose also that the function f is specified only on the space (x_i, y_j) $i = 1, \cdots, m$, $j = 1, \cdots n$, so that $f(x_i, y_j) = a_{ij}$. Let A be the matrix of function values, i.e.,

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

The problem of separating $f(x,y)$ then reduces to the problem of finding two vectors $X = (x_1, \cdots, x_m)$ and $Y = (y_1, \cdots, y_n)$ such that their outer product $X^T Y$ is as close as possible to the matrix A in the sense that it minimizes some norm $\| A - X^T Y \|$. For the definition of an abstract norm, see Ref. 6.

A. The Least-Squares or Euclidean Norm

A standard norm is the Euclidean Norm:

$$\| A - X^T Y \| = \sum_{ij} \left(a_{ij} - x_i y_j\right)^2 = E_1$$

In order to make E_1 a minimum its partials with respect to the x_i and y_j must be set to zero. Notice, however, that

$$\left(aX^T\right)\left(\frac{1}{a}Y\right) = X^TY$$

for all constants a. This means that one element of either X or Y can be fixed arbitrarily at any non-zero value. Let us, for concreteness fix $x_1 = 1$. Then the conditions to be met for minimum E_1 are

$$\left.\begin{aligned} \frac{\partial E_1}{\partial x_i} &= 0 \qquad i = 2, \cdots, m \\ \frac{\partial E_1}{\partial y_j} &= 0 \qquad j = 1, \cdots, n \end{aligned}\right\} \tag{1}$$

Equations (1) reduce immediately to

$$\left.\begin{aligned} x_i &= \frac{a_{i1}y_1 + \cdots + a_{in}y_n}{y_1^2 + \cdots + y_n^2} \qquad i = 2, \cdots, m \\ y_j &= \frac{a_{1j}x_1 + \cdots + a_{mj}x_m}{x_1^2 + \cdots + x_m^2} \qquad j = 1, \cdots, n \end{aligned}\right\} \tag{2}$$

There does not appear to be any straightforward way to solve Equations (2), but it is possible to use them as iteration equations to obtain convergence once a good guess at the solution is made.

B. The Log-Square Norm

An alternative norm which leads to a simpler computational scheme is the norm E_2.

$$||A - X^T Y|| = \sum_{ij} \log^2 \left(\frac{x_i y_j}{a_{ij}} \right) = E_2$$

The problem of minimizing E_2 can be greatly simplified by making the following substitutions:

$$u_i = \log x_i$$

$$v_j = \log y_j$$

$$b_{ij} = \log a_{ij}$$

With these substitutions we have:

$$E_2 = \sum_{ij} (u_i + v_j - b_{ij})^2$$

Taking partial derivatives of E_2 with respect to u_λ and v_μ yields:

$$u_\lambda = B_{\lambda 0} - V \tag{3a}$$

$$v_\mu = B_{0\mu} - U \tag{3b}$$

where

$$U = \frac{1}{m} \sum_i u_i$$

$$V = \frac{1}{n} \sum_j v_j$$

$$B_{\lambda 0} = \frac{1}{n} \sum_j b_{\lambda j}$$

$$B_{0\mu} = \frac{1}{m} \sum_i b_{i\mu}$$

But since $w_1 = 1$, $u_1 = \log w_1 = 0$, and by Eq. (3a) with $\lambda = 1$

$$V = B_{10}$$

Knowing V, Eq. (3a) yields

$$u_\lambda = B_{\lambda 0} - B_{10} \tag{4a}$$

Averaging the u_λ gives

$$U = B - B_{10}$$

where

$$B = \frac{1}{mn} \sum_{ij} b_{ij}$$

Knowing U, Eq. (3b) yields

$$V_\mu = B_{0\mu} - B + B_{10} \tag{4b}$$

To see that Eqs. (4a) and (4b) actually represent a solution to the system 3a and 3b notice that the average of the u_λ as calculated by Eq. (4a) is $B - B_{10}$ and that the average of the v_μ as given by Eq. (4b) is B_{10}. Substituting these values of U and V along with the values of u_λ and v_μ from Eqs. (4a) and (4b) into Eqs. (3a) and (3b) gives

$$B_{\lambda 0} - B_{10} = B_{\lambda 0} - B_{10} \tag{5a}$$

$$B_{0\mu} - B + B_{10} = B_{0\mu} - B + B_{10} \tag{5b}$$

which are identities.

The algorithm for computing the x_i and y_j from the a_{ij} is:

1. Take logarithms of the a_{ij} to get the b_{ij}
2. Average the rows and columns of the b_{ij} matrix to get the $B_{\lambda 0}$ and the $B_{0\mu}$
3. Calculate the $u_i = B_{i0} - B_{10}$
4. Calculate the average of the u_i to get U
5. Calculate the $v_j = B_{0j} - U$
6. Take anti-logs of the u_i and v_j to get the x_i and the y_j

C. Comparison of Norms

It seems natural at this point to look for some relationship between the log-square and least-square norms. Notice that if the product representation is fairly accurate, the products $x_i y_j$ will be close to the matrix elements a_{ij} so that the ratio $x_i y_j / a_{ij}$ will be close to one. But this allows the writing of

$$\log^2 \frac{x_i y_j}{a_{ij}} = \log^2 (1 + p) \tag{6}$$

where

$$p = \frac{x_i y_j}{a_{ij}} - 1$$

and is less than 1 in absolute value.

Expanding the logarithm gives

$$\log^2 \frac{x_i y_j}{a_{ij}} = p^2 - p^3 + \frac{11}{12} p^4 + \cdots$$

or

$$\log^2 \frac{x_i y_j}{a_{ij}} = \frac{(x_i y_j - a_{ij})^2 (1 - \theta p)}{a_{ij}^2}$$

where $\theta < 1$ arises from truncating the power series for $\log^2 (1 + p)$. Now let

$$a_{ij} = a + V_{ij}$$

and

$$B = a^2$$

Eq. (6) can be written

$$\log^2 \frac{x_i y_j}{a_{ij}} = (x_i y_j - a_{ij})^2 \frac{W_{ij}}{B}$$

where

$$W_{ij} = \frac{1 - \theta p}{\left(1 + \frac{V_{ij}}{a}\right)^2}$$

and

$$\theta = 1 - 1 \frac{11}{12} p + \frac{5}{6} p^2 - \frac{137}{180} p^3 + \cdots$$

Notice that if the approximation is good so that p is small and that if there is not much variation in the a_{ij} so that V_{ij} is small compared to a, then W_{ij} is close to 1. Since W_{ij} is the weighting factor which it would be necessary to include in the least-squares solution to get a result identical with the log-square method, it is seen that the two methods will yield very similar results for small p and V_{ij}.

D. Extension to Three Dimensions

Both the least-square and the log-square norms generalize easily to n dimensions. In the case of three dimensions, the norms generalize to:

$$E_1 = \sum_{ijk} (a_{ijk} - x_i y_j z_k)^2$$

$$i = 1,l \quad j = 1,m \quad k = 1,n$$

$$E_2 = \sum_{ijk} \log^2 \frac{x_i y_j z_k}{a_{ijk}}$$

The iterative equations for the least-squares solution become:

$$1. \quad x_\lambda = \frac{\sum_{jk} a_{\lambda jk} y_j z_k}{\sum_{jk} y_j^2 z_k^2}$$

$$2. \quad y_\mu = \frac{\sum_{ik} a_{i\mu k} x_i z_k}{\sum_{ik} x_i^2 z_k^2}$$

$$3. \quad z_\nu = \frac{\sum_{ij} a_{ij\nu} x_i y_j}{\sum_{ij} x_i^2 y_j^2}$$

with

$$u_i = \log x_i$$

$$v_j = \log y_j$$

$$w_k = \log z_k$$

$$b_{ijk} = \log a_{ijk}$$

and with

$$B_{\lambda 00} = \frac{1}{mn} \sum_{jk} b_{\lambda jk}$$

$$B_{0\mu 0} = \frac{1}{ln} \sum_{ik} b_{i\mu k}$$

$$B_{00\nu} = \frac{1}{lm} \sum_{ij} b_{ij\nu}$$

$$B = \frac{1}{lmn} \sum_{ijk} b_{ijk}$$

The log-square solution becomes:

$$\begin{aligned} u_\lambda &= B_{\lambda 00} - B_{100} \\ v_\mu &= B_{0\mu 0} - B_{010} \\ w_\nu &= B_{00\nu} + B_{100} + B_{010} - 2B \end{aligned}$$

E. Comparisons Between Separation and Two-Dimensional Curve Fitting

Table 1 exhibits the set of data which was used to generate the fits in Tables 2 through 5. The function values in the body of the table are the values at a point whose x-coordinate is the number of x_i at the left of the entry and whose y-coordinate is the number y_j above the entry. For example $F(0.6,1.5) = 965.18$.

Table 2 shows the results of a log-square fit to the data of Table 1. The w_i and z_j are the separating vectors, and the elements in the body of the table are the residuals $w_i z_j - F(x_i, y_j)$. The quantity E is the sum of the squares of the residuals.

Table 3 is identical in format to Table 2 and shows the results of a least-squares fit to the data. Notice that there is very little difference between the log-square residuals and the least-squares residuals. The least-squares solution *must* be better than the log-square solution in the sense that E is smaller, since the least-squares solution minimizes E whereas the log-square solution minimizes a different norm. But the log-square solution is actually better than the least-squares solution for this set of data in the *minimax* sense since the greatest absolute residual in the log-square case is 2.98 compared to 3.37 for the least-squares case.

Tables 4 and 5 compare the approximation algorithm obtained by fitting polynomials to the separating vectors of the least-squares solution with the algorithm obtained by doing a standard two-dimensional least-squares fit to the original data. In both cases the approximating equations themselves are stated at the top of the table, the body of the table consists of the residuals $\overline{F}(x_i y_j) - F(x_i, y_j)$, and E is the sum of the squares of the residuals. Note that for this set of data the product separation has a much smaller E than the two-dimensional least-squares fit even though it contains the same number (7) of parameters.

IV. THE ACCURACY PROBLEM

One of the first phenomena noticed when working with the standard least-squares process and an 8-decimal digit computing machine is that it begins to give obviously erroneous answers as the degree of the approximating polynomial approaches six. This is a discouraging phenomenon because it seems at first glance that if the function values are known to, say, four figures the coefficients determined from them should also be good to four figures. There are good reasons, however, why this should not be true.

A. Comparison of the Normal and Hilbert Matrices

If all the powers up to m are included in a one-dimensional fit, and if the x_i are fairly evenly spaced and normalized to the unit interval, a revealing approximation can be made to the resulting normal matrix. In this case

$$\sum_{i=1}^{N} x_i^p \simeq N \int_0^1 x^p \, dx = N \frac{1}{p+1}$$

and the normal matrix then looks like

$$\begin{matrix} N & \frac{N}{2} & \frac{N}{3} & \cdots & \frac{N}{m+1} \\ \frac{N}{2} & \frac{N}{3} & \frac{N}{4} & \cdots & \frac{N}{m+2} \\ \vdots & & & & \vdots \\ \frac{N}{m} & \frac{N}{m+1} & \frac{N}{m+2} & \cdots & \frac{N}{2m+1} \end{matrix}$$

But this matrix, with the N removed, is just a finite segment of the well-known Hilbert matrix which is notoriously ill conditioned (see Ref. 7).

With single precision arithmetic (8-decimal digits), matrices of this type of order greater than five cannot in general be inverted with accuracies of more than two or three digits. With double precision (17 or 18 significant digits) inverses accurate to 2 or 3 digits can be obtained for orders up to about 11. This means that, using single-precision arithmetic, curve fits of order 6 cannot be expected to improve on fits of order 5. Using double precision, fits of order 12 cannot be expected to improve on fits of order 11. This breakdown of the curve fitting process is illustrated by Figs. 1 through 4. These Figures give the error curves for various polynomial fits to the set of data of Table 6. Notice that the sum of the squares of the errors (E) is larger for the 6th-order single-precision fit than it is for the 5th-order single-precision fit, and that the same thing happens in going from order 10 to order 11 in the double precision case. It can be concluded that fits of order 6 or greater should always use double-precision arithmetic and fits of order 11 or greater cannot be done even with double-precision methods.

B. Empirically Determined Guides for Estimating the Number of Expected Coefficients

If the exponents 1 through m are not all present in the approximating polynomial or if the polynomial is in several variables, then no simple model with known inverses is available, and it is necessary to rely on experience to determine how many accurate coefficients may be obtained. Fortunately, experience is quite consistent, and fairly reliable rules can be formulated. These rules can be summarized as follows:

1. It is the order of the normal matrix that matters, *not* the order of the approximating polynomial. Therefore, it is necessary to think of the number of coefficients which can be obtained rather than the order of the polynomial.

2. With single precision (8-decimal digits) in one-dimension, approximately 6 "good" coefficients can be expected. (By good is meant that E for the fit will be smaller than E for the fit with one less coefficient).

3. The ratio of precisions in the arithmetic multiplies the number of expected good coefficients (i.e., double precision can be expected to give twice as many good coefficients as single precision).

4. The number of dimensions is a power on the number of expected good coefficients (e.g., the number of expected good coefficients in a single precision three-dimensional fit is 6^3 and the number of expected good coefficients in a double precision two-dimensional fit is $(2 \times 6)^2$.

Normalization of all the variables either to the interval (0,1) or to the interval (−1,1) should always be done before curve fitting. This normalization causes the normal matrix to be a little further from singular and the resulting coefficients to be near 1.

V. CURVE FITTING ROUTINES AVAILABLE AT THE JET PROPULSION LABORATORY

The Datatron least-squares program accepts up to 200 fixed decimal values for the dependent and independent variables. These numbers are not destroyed by the operation of the program, so they may be re-used without being re-entered. The user specifies the order fit desired by entering the order in a fixed memory location and chooses single- or double-precision arithmetic by the setting of a switch on the computer console. The program will not fit polynomials of order greater than 6 when operating in the single-precision mode and will not fit polynomials of order greater than 12 under any conditions. One of the advantages of this routine is that if a single-precision fit is done and turns out not to be sufficiently accurate, the switch on the console can be flipped down, putting the computer in double-precision mode, and the problem can be immediately re-run to obtain the double precision fit using 24 x,y pairs. This routine takes 3.5 min for a fourth order single-precision fit and 10.5 min for a double-precision fourth order fit. Sixth order fits take 6.5 min and 16 min for single- and double-precision, respectively. A twelfth order fit which must be done double-precision takes 45 min.

Also available is a separation program for the 704. This program accepts as input a matrix a_{ij}, $i = 1, m$, $j = 1, n$. The a_{ij} must all be positive. The user has the option of obtaining either a log-square separation or both the log-square and least-squares solutions. His choice is indicated by a control card which is read in with the data cards containing the a_{ij}. In either case the output consists of the separating vectors u_i and v_j, the error matrix $(a_{ij} - u_i v_j)$, and the error

$$E_1 = \sum_{ij} \left(a_{ij} - u_i v_j\right)$$

In the event that both log-square and least-squares solutions are taken, it will be noticed that E is larger for the log-square solution than for the least-squares solution. The reason is that the log-square code minimizes

$$E_2 = \sum_{ij} \log^2 \left(\frac{u_i v_j}{a_{ij}} \right)$$

not

$$E_1 = \sum_{ij} \left(a_{ij} - u_i v_j\right)^2$$

but it is E_1 which is listed for both solutions. This code will fit any matrix of dimensions less than or equal to 20 × 20. It requires approximately 15 sec to fit a 7 × 13 matrix using the log-square norm and another 30 sec to calculate a least-squares solution.

The two-dimensional least-squares program written at JPL for the 704 fits a polynomial of the form $P(x,y) = a_0 x^{\lambda_0} y^{\mu_0} + \cdots + a_m x^{\lambda_m} y^{\mu_m}$ to a set of data x_i, y_j, F_{ij} $i = 1, m$, $j = 1, n$ where $F_{ij} = F(x_i, y_j)$. This is a single-precision program, so at most 36 coefficients can be expected from a fit using this program. There are no restrictions on the choice of the λ_i, μ_i pairs. The output from this program consists of a listing of the λ, μ pairs with the associated coefficients beside them, a listing of the matrix of the values of $P(x,y)$ and $F(x,y) - P(x,y)$, and the sum of the squares of the errors E. The function values of the data for this program are restricted to lie on a lattice of dimensions less than or equal to 20 × 20 and to determine the coefficients of a polynomial with 10 coefficients over this full space would require approximately 40 sec.

The three-dimensional least-squares program written at JPL for the 704 fits a polynomial of the form $P(x,y,z) = a_0 x^{\lambda_0} y^{\mu_0} z^{\nu_0} + \cdots + a_m x^{\lambda_m} y^{\mu_m} z^{\nu_m}$ to a set of data x_i, y_j, z_k, F_{ijk}, $i = 1, l$, $j = 1, m$, $k = 1, n$ where $F_{ijk} = F(x_i, y_j, z_k)$. Space and time limitations impose a restriction to 60 coefficients when using this program. There are no restrictions on the choice of the (λ, μ, ν) triplets. The output of this routine consists of a listing of the (λ, μ, ν) triplets with the associated coefficients beside them, a listing of the matrix of the errors, and the sum of the squares of the errors E. The data points for this program are required to lie on a lattice 5 × 5 × 13. To determine 20 coefficients over such a lattice takes approximately 3 min.

A general least-squares curve fit routine for multidimensional work is now being written. This routine will fit polynomials in either two or three dimensions with either single or double precision and will carry along a weighting factor W so as to minimize $\Sigma W (P - F)^2$ where $W \geq 0$. The points used do not have to be equally spaced or to lie on a lattice, and any combination of exponents may be used.

VI. CONCLUSIONS

The techniques discussed have been useful in curve-fitting analysis using a general purpose digital computer. The exponent-addition-table way of looking at the structure of the normal matrix greatly simplified the coding of two- and three-dimensional curve-fitting programs and caused the resulting programs to be much more flexible than they would otherwise have been. The separation theory has supplied a rather simple algorithm for performing a calculation which had been in the past rather cumbersome. All of the routines written as a result of these investigations have been extensively used already and will probably continue to be heavily used in the future.

Table 1. Original Data for Separation Comparisons

X	Y						
	0.0	0.5	1.0	1.5	2.0	2.5	3.0
0.0	398.11	876.42	1198.61	1376.01	1400.18	1275.19	1000.71
0.1	285.46	623.69	857.23	979.39	998.07	908.13	712.92
0.2	218.72	478.46	655.46	752.87	766.19	697.63	546.11
0.3	193.93	423.03	579.42	661.64	673.95	614.82	483.82
0.4	198.65	439.45	598.22	687.30	701.42	636.09	502.22
0.5	230.37	505.81	697.89	797.31	812.02	738.42	580.01
0.6	282.42	617.43	843.29	965.18	982.91	895.45	702.19
0.7	337.32	741.93	1018.05	1168.64	1189.53	1082.11	848.04
0.8	399.17	873.57	1199.85	1374.65	1398.63	1275.54	999.24
0.9	456.68	992.52	1360.33	1558.82	1588.73	1447.85	1135.03
1.0	492.98	1078.86	1480.71	1697.87	1728.12	1573.83	1235.16
1.1	510.40	1121.69	1536.08	1758.27	1790.82	1631.86	1279.01
1.2	499.03	1093.78	1497.91	1719.26	1751.23	1594.80	1249.31

Table 2. Log-Square Separation for Data of Table 1

(E = 139.08)

W	Z						
	399.40	875.25	1199.51	1374.40	1399.75	1274.44	1000.11
1.00000	1.29	-1.17	0.90	-1.61	-0.43	-0.75	-0.60
0.71328	-0.57	0.61	-1.64	0.95	0.34	0.91	0.44
0.54705	-0.23	0.34	0.72	-1.01	-0.47	-0.45	1.00
0.48300	-1.02	-0.29	-0.06	2.19	2.12	0.73	-0.77
0.50009	1.08	-1.74	1.63	0.02	-1.42	1.25	-2.08
0.57944	1.06	1.34	-2.85	-0.93	-0.95	0.04	-0.50
0.70353	-1.43	-1.66	0.60	1.76	1.86	1.16	1.43
0.84830	1.50	0.54	-0.51	-2.74	-2.13	-1.00	0.35
0.99959	0.07	1.32	-0.84	-0.81	0.54	-1.62	0.47
1.13594	-2.98	1.71	2.24	2.42	1.30	-0.16	1.04
1.23446	0.06	1.61	0.03	-1.22	-0.19	-0.58	-0.56
1.27972	0.72	-1.61	-1.04	0.59	0.47	-0.92	0.86
1.25006	0.25	0.34	1.55	-1.17	-1.46	-1.67	0.89

Table 3. Least-Squares Separation for Data of Table 1

(E = 104.80)

W	Z						
	399.58	875.40	1199.82	1375.15	1400.35	1275.41	1000.22
1.00000	1.48	−1.02	1.21	−0.86	0.17	0.22	−0.49
0.71278	−0.64	0.28	−2.01	0.79	0.07	0.96	0.02
0.54687	−0.20	0.27	0.69	−0.84	−0.38	−0.14	0.88
0.48218	−1.26	−0.94	−0.89	1.43	1.26	0.15	−1.54
0.50005	1.16	−1.79	1.75	0.34	−1.17	1.69	−2.06
0.57974	1.28	1.69	−2.30	−0.08	−0.18	0.99	−0.14
0.79248	−1.72	−2.48	−0.43	0.84	0.81	0.50	0.45
0.84875	1.83	1.06	0.30	−1.48	−0.99	0.40	0.90
0.99936	0.16	1.27	−0.79	−0.38	0.82	−0.94	0.34
1.13444	−3.37	0.56	0.80	1.20	−0.12	−0.97	−0.34
1.23413	0.16	1.49	0.03	−0.75	0.09	0.20	−0.75
1.27931	0.79	−1.78	−1.13	0.98	0.66	−0.21	0.59
1.24982	0.38	0.31	1.66	−0.56	−1.04	−0.76	0.79

Table 4. Product Fit for Data of Table 1

(E = 110.86)

$$\overline{F}(x,y) = (1.00002 - 3.5418x + 7.0289x^2 - 3.2531x^3)(399.74 + 1100.5y - 300.10y^2) = \overline{F}_1(x)\,\overline{F}_2(y)$$

x	$F_1(x)$	$y = 0.0$ $F_2(y) = 399.74$	$y = 0.5$ $F_2(y) = 874.96$	$y = 1.0$ $F_2(y) = 1200.12$	$y = 1.5$ $F_2(y) = 1375.23$	$y = 2.0$ $F_2(y) = 1400.30$	$y = 2.5$ $F_2(y) = 1275.31$	$y = 3.0$ $F_2(y) = 1000.27$
0.0	1.00002	1.64	−1.44	1.53	−0.75	0.15	0.15	−0.42
0.1	0.71287	−0.50	0.04	−1.70	0.97	0.16	1.00	0.14
0.2	0.54678	−0.15	−0.05	0.74	−0.92	−0.53	−0.32	0.82
0.3	0.48224	−1.16	−1.09	−0.67	1.55	1.33	0.19	−1.45
0.4	0.49972	1.11	−2.21	1.50	−0.07	−1.66	1.21	−2.37
0.5	0.57970	1.36	1.40	−2.18	−0.09	−0.27	0.88	−0.15
0.6	0.70266	−1.54	−2.63	−0.01	1.14	1.02	0.66	0.66
0.7	0.84909	2.10	0.99	0.96	−0.95	−0.55	0.74	1.28
0.8	0.99946	0.35	0.92	−0.38	−0.16	0.91	0.92	0.49
0.9	1.13444	−3.20	0.07	1.13	1.30	−0.17	−1.09	−0.28
1.0	1.23413	0.35	0.95	0.39	−0.66	0.03	0.07	−0.70
1.1	1.27931	0.99	−2.34	−0.75	1.08	0.60	−0.34	0.65
1.2	1.24982	0.57	−0.24	2.02	−0.47	−1.11	−0.89	0.85

Table 5. Two-Dimensional Curve Fit for Data of Table 1

$(E = 2.334.10^6)$

$$F(x_1y) = 283.26 + 937.66y - 651.36x - 168.92xy + 908.31x^2 - 260.14y^2 + 235.27x^2y$$

X_i	Y_i						
	0.0	0.5	1.0	1.5	2.0	2.5	3.0
0.0	-114.84	-189.36	-237.83	-271.58	-282.17	-273.66	-245.74
0.1	-58.24	0.04	32.96	47.18	34.82	1.00	-57.62
0.2	-29.39	102.47	187.00	221.06	209.14	149.02	41.80
0.3	-24.32	135.62	238.20	284.88	271.40	199.29	68.98
0.4	-30.60	117.44	217.42	257.03	241.53	175.42	47.76
0.5	-45.71	69.82	138.64	170.06	156.11	100.40	-0.57
0.6	-62.98	-2.52	37.01	50.45	37.98	0.62	-61.00
0.7	-64.93	-67.23	-71.11	-79.53	-88.31	-98.86	-112.83
0.8	-55.67	-118.56	-163.40	-186.83	-189.50	-175.18	-137.72
0.9	-23.90	-136.68	-211.50	-247.06	-244.12	-200.46	-114.92
1.0	47.23	-101.67	-196.62	-236.96	-220.45	-149.47	-24.18
1.1	155.42	-2.64	-93.87	-122.97	-92.51	-0.60	155.13
1.2	310.57	187.66	125.30	115.64	165.30	273.29	440.26

Table 6. Original Data for Figs. 1 through 4

X	Y
0.0300	52.110
0.0500	56.206
0.0700	59.209
0.1000	62.189
0.1200	63.617
0.1500	65.808
0.1700	66.973
0.2000	68.439
0.2200	69.267
0.2500	70.449
0.2700	71.192
0.3000	72.130
0.3200	72.685
0.3500	73.558
0.3700	74.138
0.4000	74.940
0.4200	75.443
0.4500	76.181
0.4700	76.661
0.5000	77.370
0.5200	77.835
0.5500	78.526
0.5700	78.980
0.6000	79.640

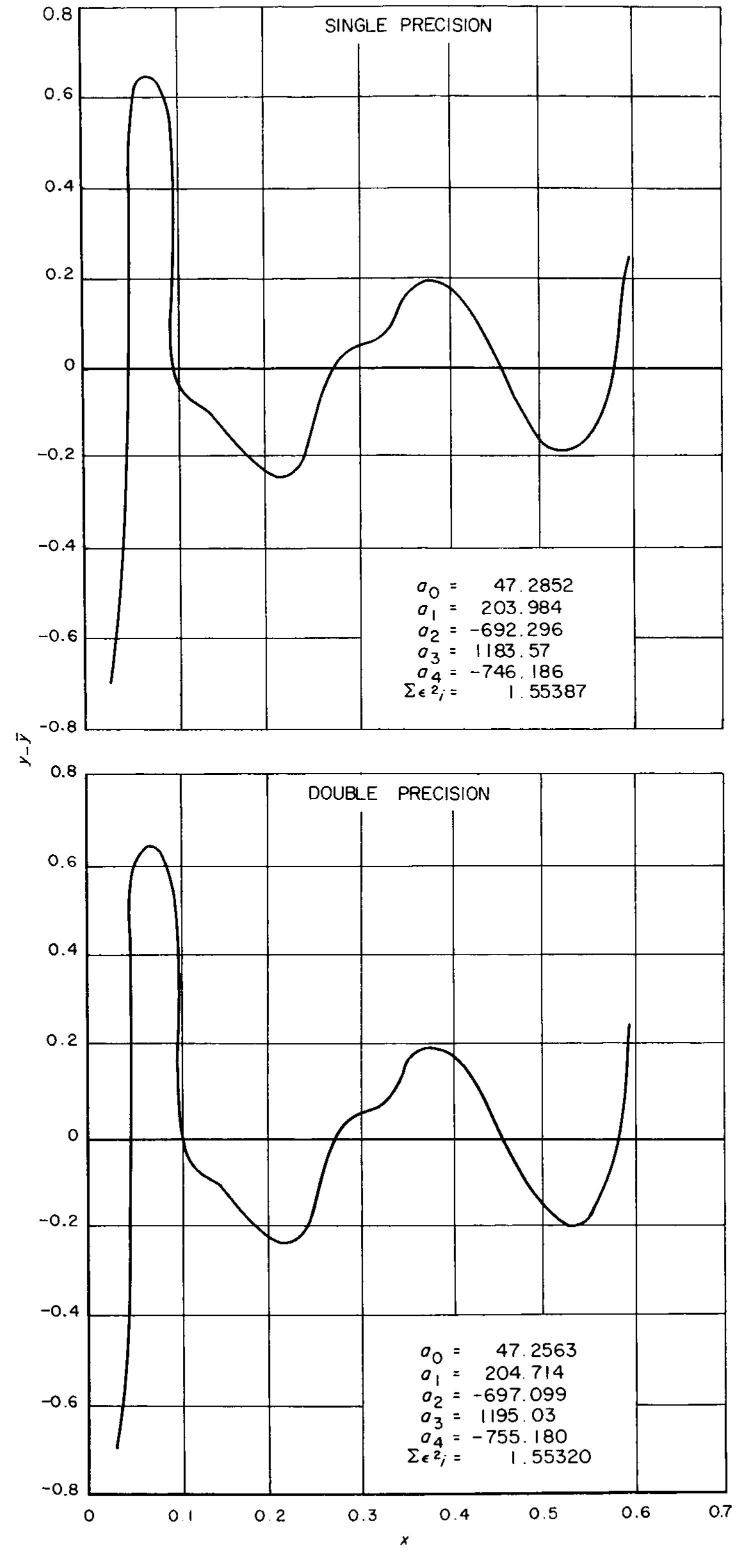

Fig. 1. Comparison of Single and Double Precision Curve Fits of Fourth Order for Data of Table 6

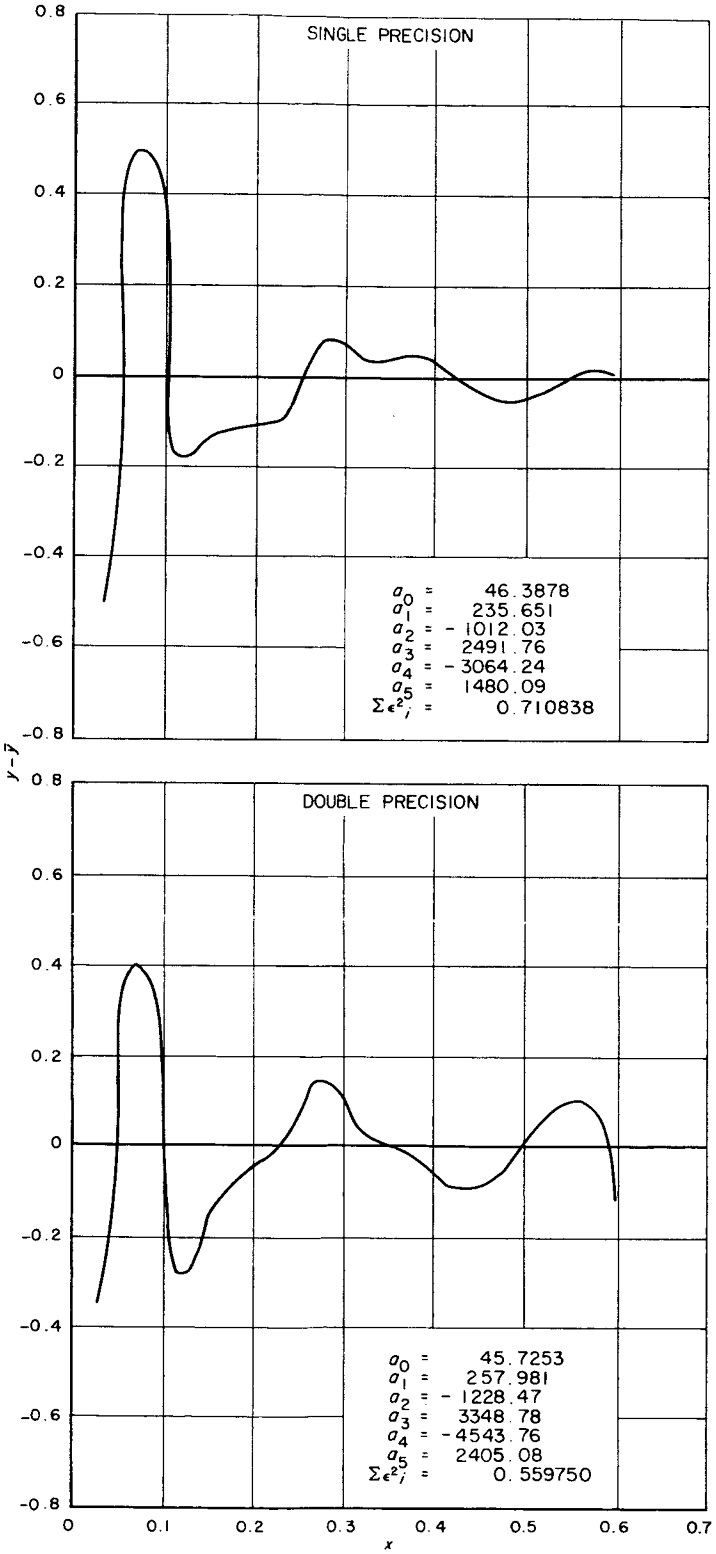

Fig. 2. Comparison of Single and Double Precision Curve Fits of Fifth Order for Data of Table 6

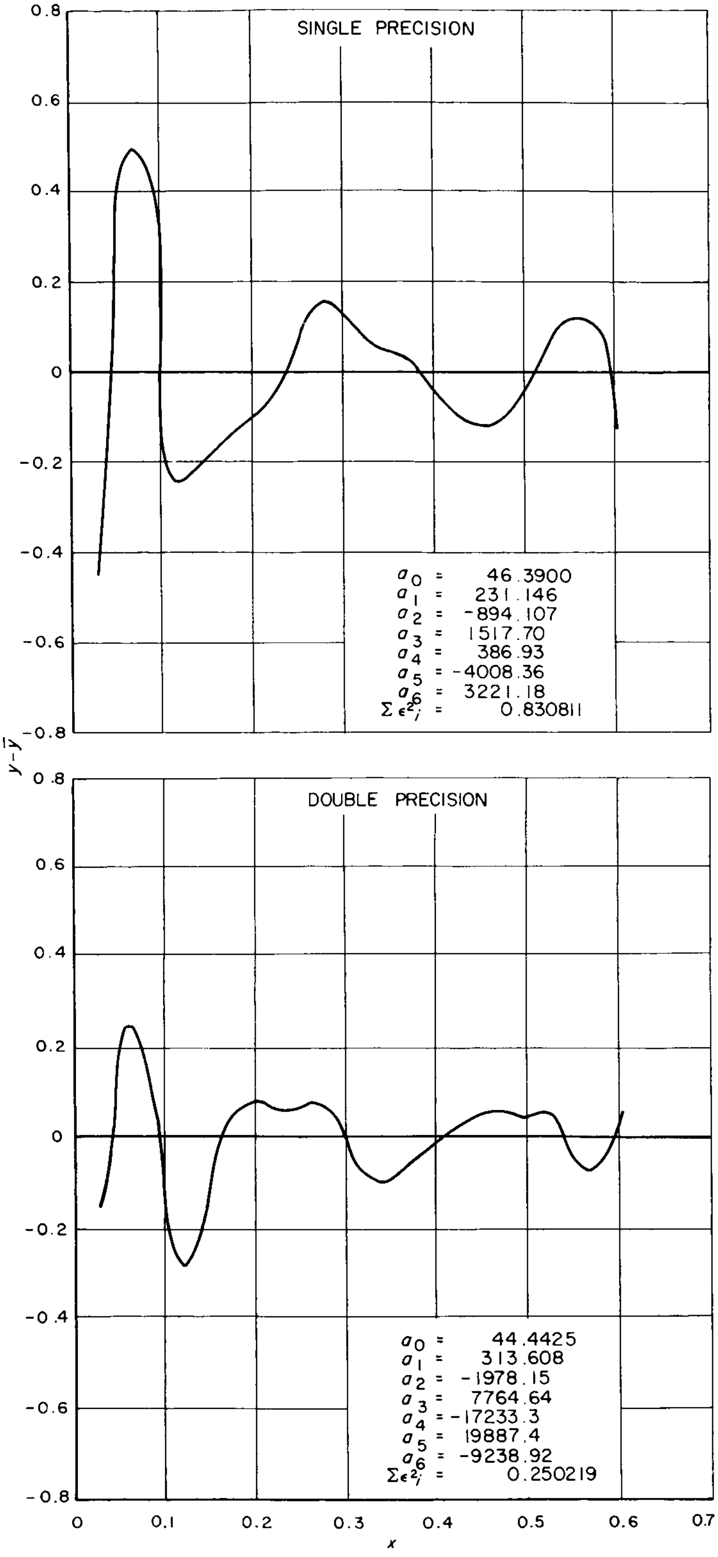

Fig. 3. Comparison of Single and Double Precision Curve Fits of Sixth Order for Data of Table 6

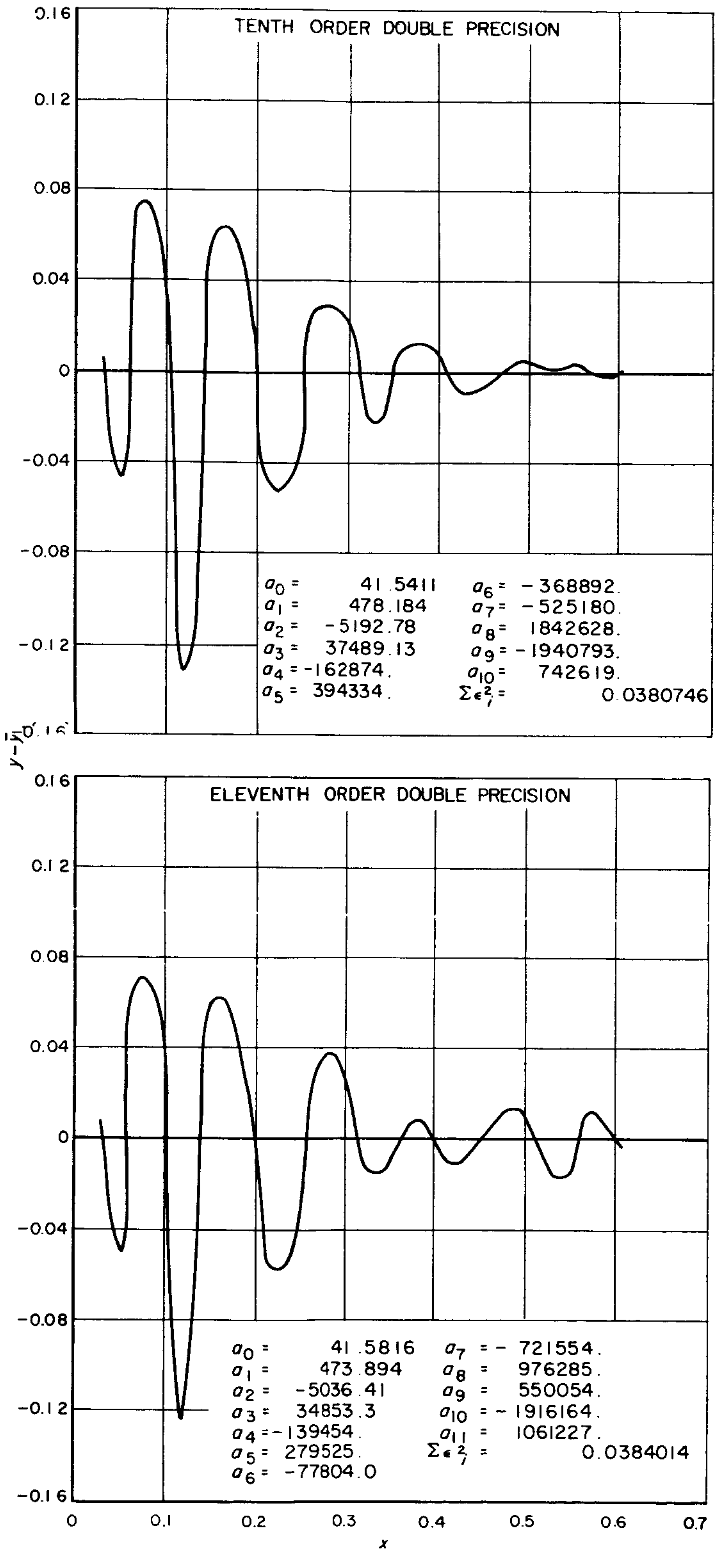

Fig. 4. Comparison of Tenth, Eleventh, and Twelvth Order Double Precision Curve Fits for Data of Table 6

REFERENCES

1. Hald, A., *Statistical Theory with Engineering Applications*, New York: John Wiley and Sons, 1952.

2. Hastings, C., *Approximations for Digital Computers*, Princeton: Princeton University Press, 1955.

3. Householder, A. S., *Principles of Numerical Analysis*, New York: McGraw Hill Book Company, 1953.

4. Forsythe, G. E., *Generation and Use of Orthogonal Polynomials for Data Fitting with a Digital Computer*, J. Soc. Indust. Appl. Math., Vol. 5, No. 2, June 1957.

5. Hoel, P. G., *Introduction to Mathematical Statistics*, New York: John Wiley and Sons, 1951.

6. Munroe, M. E., *Introduction to Measure and Integration*, Cambridge (Mass.): Addison Wesley Publishing Company.

7. *Contributions to the Solution of Systems of Linear Equations and the Determination of Eigenvalues*, National Bureau of Standards Applied Mathematics Series 39, September 30, 1954.

APPENDIX A

Least Squares and Maximum Likelihood

Given

1. A set of x_i, $i = 1, n$
2. An mth-order polynomial $P_m(X)$
3. The value of σ

A set of y_i has been generated as follows:

$$y_i = P_m(x_i) + N_i(0,\sigma)$$

where the N_i are independent samples of a normally distributed variable with mean zero and variance σ^2.

This data set is now a model for an empirically determined set of noisy data in which the assumption is made that:

1. The underlying function is an mth-degree polynomial.
2. The noise is uncorrelated random noise, normally distributed with mean zero and variance σ^2.

Now suppose that we have been given m, and the exact values of the x_i and y_i and are required to find the a_i, $i = 0, \cdots, m$ which maximize the likelihood of our sample. The likelihood of a sample y_i, $i = 1, \cdots, n$ where y has the distribution function f is defined to be

$$\prod_{i=1}^{n} f(y_i)$$

In our case the variable which is normally distributed is $y_i - P_m(x_i)$ and f is

$$\frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x-m}{\sigma}\right)^2\right]$$

The likelihood of our sample is

$$L = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y_i - P_m(x_i)}{\sigma}\right)^2\right]$$

or

$$L = (\sigma\sqrt{2\pi})^n \exp\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{n}[y_i - P_m(x_i)]^2\right]$$

Instead of maximizing L

$$\operatorname{Log} L = -n \log \sigma - \frac{n}{2} \log 2\pi - \frac{1}{2\sigma^2} \sum_{i=1}^{n} [y_i - P_m(x_i)]^2$$

is maximized. The parameters available for maximization are σ and the

$$a_i, \; i = 0, \cdots, m$$

Partialling on σ yields

$$\frac{\partial L}{\partial \sigma} = \frac{n}{\sigma} + \frac{1}{\sigma^3} \sum_{i=1}^{n} [y_i - P_m(x_i)]^2$$

Setting this partial equal to zero yields

$$\frac{n}{\sigma} + \frac{1}{\sigma^3} \sum_{i=1}^{n} [y_i - P_m(x_i)]^2 = 0$$

or

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} [y_i - P_m(x_i)]^2$$

which is the standard minimum variance estimator. Partialling on the a_i yields

$$\frac{\partial L}{\partial a_\nu} = -\frac{1}{\sigma^2} \sum_{i=1}^{n} [y_i - P_m(x_i)]\, x_i^\nu \qquad \nu = 0, \cdots, m$$

Setting these partials equal to zero yields the standard normal equations of least squares. Thus it can be seen that under the assumptions of (1) underlying mth order polynomial and (2) uncorrelated normally distributed noise with zero mean, the least-squares procedure is a maximum likelihood procedure.

APPENDIX B

Generation of Orthogonal Polynomials

Write the recursive equations as follows:

1. $P_{-1}(x) = 0$
2. $P_0(x) = 1$
3. $P_{n+1}(x) = xP_n(x) - \alpha_n P_n(x) - \beta_{n-1} P_{n-1}(x)$

The proof that it is possible to choose α_n and β_{n-1} to obtain a set of orthogonal polynomials proceeds by induction. Assume that there already is a set $P_0, P_1, \cdots, P_k$ of polynomials such that $(P_\mu, P_\nu) = 0$ if $\mu \neq \nu$ and look at

$$P_{k+1} = xP_k - \alpha P_k - \beta P_{k-1}$$

Notice that P_{k+1} is already orthogonal to P_α, $\alpha < k - 1$ since

$$(P_{k+1}, P_\alpha) = (xP_k - \alpha P_k - \beta P_{k-1}, P_\alpha)$$

$$= (P_k, xP_\alpha) - (P_k, P_\alpha) - (P_{k-1}, P_\alpha)$$

but xP_α is a polynomial of degree less than k, so that $(P_k, xP_\alpha) = 0$, and P_α is of degree less than $k - 1$, so that $(P_k, P_\alpha) = (P_{k-1}, P_\alpha) = 0$.

Therefore, all that is needed is to make P_{k+1} orthogonal to P_k and P_{k-1} by proper choice of α and β. But

$$(P_{k+1}, P_k) = (xP_k - \alpha P_k - \beta P_{k-1}, P_k)$$

$$= (xP_k, P_k) - \alpha(P_k, P_k) - \beta(P_k, P_{k-1})$$

and

$$(P_{k+1}, P_{k-1}) = (xP_k - \alpha P_k - \beta P_{k-1}, P_{k-1})$$

$$= (xP_k, P_{k-1}) - \alpha(P_k, P_{k-1}) - \beta(P_{k-1}, P_{k-1})$$

If these two inner products are set to zero there results:

$$(xP_k, P_k) - \alpha(P_k, P_k) = 0$$

$$(xP_k, P_{k-1}) - \beta(P_{k-1}, P_{k-1}) = 0$$

These equations have the following solutions

$$\alpha = \frac{(xP_k, P_k)}{(P_k, P_k)}$$

$$\beta = \frac{(xP_k, P_{k-1})}{(P_{k-1}, P_{k-1})}$$

Thus the induction is complete and can be carried to the point where

$$(P_k, P_k) = 0$$

But $(P_k, P_k) = 0$ implies $P_k(x_i) = \dot{0}$, $i = 1, \cdots, n$, and this cannot occur until $k = n$ since it takes an nth degree polynomial to have n roots, and since the orthogonal polynomials are guaranteed to be nontrivial by the fact that their leading coefficient is always unity.

This analysis assumes no special properties of the inner product used here and can be carried through in an entirely analogous manner with a general inner product.